Mastering Specification Heterogeneity with Multifacet Analysis

Christian Attiogbe

Mastering Specification Heterogeneity with Multifacet Analysis

J. Christian Attiogbé
LINA - UMR CNRS 6241 - Université de Nantes
Christian.Attiogbe@univ-nantes.fr

Abstract

We address the specification and the formal analysis of systems involving multiple facets. This leads to heterogeneous specifications that cover the different facets of these systems. A method, multifacet specification and analysis, is proposed to master the heterogeneity of systems by considering an abstract reference model that is refered to as a semantic reference for the specific models which are built from the reference one. Experiments achieved with Mobile Ad-hoc Network system and the obtained results are presented.

1. Introduction

We address the specification and the formal analysis of systems involving multiple facets. This leads to heterogeneous specifications that cover the different facets of these systems. A facet of a system is viewed, either according to its specification or according to its analysis. Data abstraction or communication with the environment are examples of facets of a system. Correctness with respect to specific properties, behavioural properties, time-constraints, are other examples of facets related to the analysis of a system. Therefore *multifacet* refers to the various features that appear in a system. Accordingly, the data facet and the related properties, the control facet and the related facets, nonfunctional facets, etc may be studied separately and with appropriate formal models (or formal specifications) and tools. Hence the specification heterogeneity that leads to difficult issue with respect to formal analysis and mechanization.

Motivation. Our motivation is to avoid the impasse of an *à posteriori* heterogeneous formal analysis of a system by ensuring the consistency of the various parts of the system during their construction.

In the practice, in order to capture the various facets that appear in a system requirement (or a problem statement), one solution is to use the appropriate languages, techniques and tools for the identified parts of the global system. The different specifications related to the various languages being integrated to model the global system; hence an heterogeneous specification of the system.

As far as the formal analysis of the global system is concerned, the various parts may be separately analysed according to the properties expressed on them; it is also possible either to translate the various specifications into a logical reasoning framework or to translate them into the input language of a general purpose tool where the properties are expressed and analysed. These approaches are not efficient. The use of several languages, techniques and tools often leads to difficulties in formal analysis of the global system. The techniques and tools used for formal analysis require specific input formal models expressed with particular languages. The global system may be inconsistent if there is no insurance that the models used for each specific analysis are consistent (or even equivalent). On the other hand, translating independent specifications into the input language of a general purpose tools is not always possible (there may be some incompatibilities); when the translation is possible it should be established that the semantics are preserved; however there may be loss of information. Therefore covering the different aspects of a system with various languages and analysis techniques is a challenging concern.

Contribution. The approach that we propose to fight the heterogeneity concern is to perform *multifacet specification and analysis* of a system by considering an *abstract reference model* that is refered to as a semantic reference for the *specific models* which are built from the reference one and which are dedicated to the different facets or to the subsystems to be analysed.

A specific formal model is the one that focuses on particular features of a system instead of embrassing the global features of the system at hand. A specific model may be well suited for a given subsystem. The reference model on the other hand sets the common requirements, the common constraints and the global properties of the system at hand.

We show in this work how one can build specific models from a global one, how to reason on the global system,

how to maintain the consistency of the global system with respect to the modifications on the specific models. That leads to mastering the system heterogeneity.

We illustrate the proposals with an example of multifacet specification using Event B [2, 4] as practical framework and combining theorem-proving and model-checking tools for the multifacet analysis.

The article is structured as follows. In Section 2 we present the problems raised by heterogeneous specifications and motivate the multifacet approach. In Section 3, we present the multifacet specification and analysis approach. Section 4 deals with the illustration of the approach on the MANET case study. We conclude the article with the Section 5 where we give some perspectives of the current work.

2. Specification Integration and Limitations

2.1. Integration of Heterogeneous Specifications

The heterogeneous specifications context is as follows. Well-suited specification languages are used to express parts of a global system; more specifically, regarding property analysis perspective, the input languages of analysis tools are selected to express the specification of parts of the global system.

Reasoning on the system is considered from two points of view. First, reasoning is performed by expressing and proving properties on the various specifications. This approach is definitely insufficient since one cannot deduce the properties of the global system from the properties proved on subsystems. This is due to the fact that the specification of the parts are not semantically linked and they may use different logics.

Second, a more widely used approach is to prove properties using a dedicated environment after the translation of the specifications into the input format of this environment. But we still cannot connect the different parts previously specified and analysed in an independent way.

The translation of specifications is one of the main solutions used for heterogeneous specifications and their integration; it is based on embedding techniques [9]. A translation may be viewed at high level where the semantics is preserved at the language level (the source language of a specification is linked with the target language); it may also be viewed at specification level where only the given specification is translated into an equivalent specification.

Therefore, the translation from one formal model to another one results in relating equivalent concepts of both models. This implies that we have a *compatibility* between the considered formal models.

Definition 1 (Model Compatibility) *A model M_1 and a model M_2 are compatible if they are built on the same semantic basis*[1].

2.1.1. Basis of Translation The main concepts or units involved in the translation between models are:

- Data-oriented concepts: type, variable, term or expression, formula, predicate, relation;
- Behaviour-oriented concepts: action, behaviour, process, communication, synchrony and asynchrony, process composition, sequencing or prefixing, choice of behaviour, interleaving, parallelism and non-determinism;
- Transversal concepts: they are module and time. Subsystem (or component) is a coarse grain of module, it embodies data or processes.

When translating specifications, a matching between the concepts is considered and the translation may be straightforward if the semantics of the considered specifications are very close.

2.1.2. Lessons from Concepts Translation According to the formal specification of process behaviours, going from an asynchrony context to a synchrony context is manageable. For example, to deal with synchrony hypothesis going from asynchronous formal models based on finite state machines, epsilon-transitions are added to the states in order to enable synchronous evolution with the other process states. Unlikely, it is not the case when going from synchronous context to an asynchronous one; hence there is incompatibility when dealing with such heterogeneous specifications.

Reducing the complexity of infinite state space (infinite system) is manageable by considering data abstraction and system/process behaviour abstraction. Data abstraction may reach information and property loss.

2.2. Formal Analysis

The standard way to perform formal analysis of a system S which has desired properties P, is to build a mathematical model M of S and then to ensure that M has the properties P (that is $M \models P$).

Regarding discrete event systems for example, there are several specification methods: state machines, Petri nets, action systems, Event B, etc. Consider the specification and the analysis of a given system with one of these methods; according to the selected method and its related tools, some properties (P_s) may be expressed and analysed and other properties may not. In all the cases we shall establish: $M \models P_s$. However, it will remain to discuss $M \models P$.

1 For instance a logic, a finite state model, etc

2.3. Limitations of Formal Analysis

The analysis of various properties on a given system in a heterogeneous context results in building the models M_i, the properties P_i and then establishing:

$$M_1 \vDash P_1 \qquad M_2 \vDash P_2 \qquad M_3 \vDash P_3 \qquad \cdots$$

But nothing can be said about the global system and the relationship between the M_i models. Accordingly a solution might be the translation of the M_i as described previously and the translation of the properties P_i. But, the main problem (known as the *transfer problem*) is that in a general setting, a logical property cannot always be transfered from a logical system to another one. In the current case, a given specific model (M_i) and its related properties (P_i) in the selected framework, act as a logical system. Consequently there are limitations on the formal analysis of the global system made of the combination of (M_i, P_i).

2.4. Solutions to the Analysis

A solution to the previous limitations related to formal reasoning is to examine thoroughly the relationship between the M_i models and then to conclude under which hypothesis the analysed properties hold for the global system.

From the practical point of view, it is not generally feasible to compare formal models. But in cases such as the use of automata (finite state machines), one can establish the equivalence of automata[2], or the bissimulation[17, 19] between processes in the case of process models. In general, one has to find the hypothesis for deciding if two or more models with their properties have some commonalities. The question of compatibility between models is raised again. When the models are compatible, one can embed them within a semantics basis and study step by step the range of solutions for shared properties. Whatever the case, in a general setting, the transfer problem will be the last limitation of this approach: it is not practically tractable to compare models and to relate their properties.

For all these reasons, combining specifications afterwards is not a good practice with respect to the analysis of multifacet systems. This motivates our proposal of the multifacet analysis as a solution which is more practical and efficient: it consists to derive specific models from an abstract reference model. The main idea is to build models which are *à priori* consistent by construction, instead of trying *à posteriori* to build a relation between the independent models.

3. The Multifacet Approach

We have introduced in [6] an approach to analyse a system using an unique formal model as a *reference* to specific analysis techniques whose input formats can be different.

3.1. Overview of the Approach

Our proposal to specify and analyse a given system consists (see Fig. 1):

1. to build an abstract formal model from the system at hand and to state the desired global properties according to this formal model; it is the *reference model*;
2. to systematically derive or translate from this abstract model, other formal models which are specific to various analysis techniques;
3. to perform analysis (verification of properties) with the specific models or with their extensions, by adding specific properties to the global ones;
4. to ensure the consistency between the reference model and the specific ones by propagating the feedback from the specific models study on the reference model and by updating consequently the other specific models. Then, the analysis of each facet via a specific model participates in the global system analysis.

Regarding a development process, one of the specific models may be well-suited to code generation; the other ones have being used for instance to simulate the intended system, to analyse correctness properties, etc.

For example, in a given development project a Petri net [18] may be used to simulate the system, perform reachability analysis whereas a B model [4] may be used to perform successive refinement to code.

The reference model may be an abstract mathematical model, in FOL for instance, or a model expressed with an ad-hoc specification language. The latter solution seems restrictive according to the syntax and semantics of the chosen language; however unlike with the abstract mathematical model, the restrictions may make it easy the construction of efficient specific models.

Proposition 1 (Model Derivation) *It is necessary to derive specific models from a reference basic model to ensure the consistency of multifacet analysis.*

The specific models may be or not homomorphic to the reference model; but it should be possible to modify the reference model to take into account the feedback from the specific models (see Fig. 1).

3.2. Building the Reference Model: Principle

The main feature of the reference model is that it constitutes a global abstraction of the system at hand independently from any analysis technique or tool. Therefore it should be built at an appropriate abstraction level that

2 with preorders, simulations for example

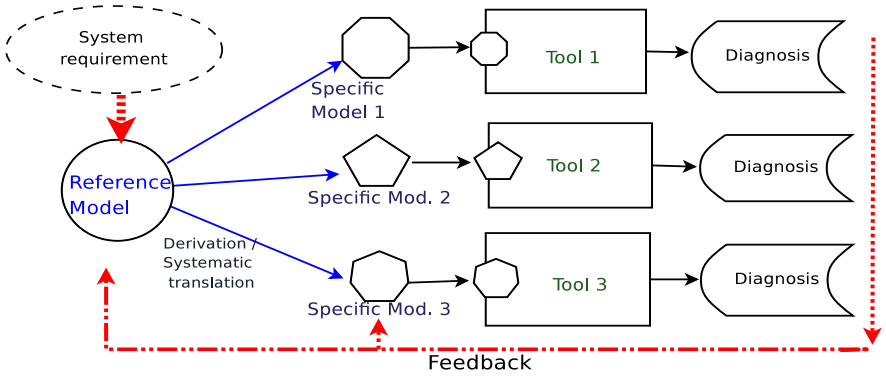

Figure 1. Schematic principle of multifacet analysis

permits the expression of the system global properties. Depending on the system nature, one might use either a first order logic and state invariant properties, temporal properties, etc; or a state space graph and express accessibility properties, etc. At least, to get a global model ready to allow facets derivation and verifications, one needs a data part and a dynamic or behavioural part in the high level abstract model. These parts may be extended later during the specification.

As far as the specific models are concerned, on the one hand their derivation depends on the targeted tools and their related input languages and on the other hand it depends on the compatibility between the reference model and the specific ones with respect to the tradeoffs to preserve properties.

When deriving a specific model from the reference one, information may be lost, added or remained the same. Accordingly, in order to conduct the analysis with a consistent specific model, we should know which global properties may be preserved and which one may be lost. For example it is often the case, to reduce the space of values of a system in order to use graph exploration techniques (model-checking tools); this is generally done to reduce infinite systems into finite ones in order to use the analysis tools associated to the latter.

However the tradeoffs made here are that getting an error-free reduced system after its analysis, does not mean that the initial system is correct; but when errors are detected on the reduced system, it is sure that the initial system is also incorrect and then it should be corrected.

3.3. Specific Models and Property Analysis

The specific models are built by *translation*, by *reduction*, by *extension* from the reference model.

As far as the translation is concerned, generally, one should preserve the semantics of the translated model and its global properties. Bissimulation techniques [17, 19] may be used to establish the desired semantic and property preservation.

When one reduces the data part of a model during the reduction process, the properties of the model cannot always be transfered to the model resulting from the reduction. For instance only the dynamic part of a system is checked when the analysis is performed on a system where the data part is completely abstracted (that is the reduction of a data-parameterized system to a system without data).

When the reference model is extended (to cover other facets) during the derivation process to obtain a more expressive specific model, the interpretation of the properties of the specific model needs some care; the properties established on the specific model are transfered to the reference model but properties not established (not proved) on the specific model are not necessarily preserved for the reference model; it may be the case of local properties which are meaningful for the specific model but not for the reference one. Accordingly it is important to distinguish between local properties and global ones. Therefore local properties may be considered as complementary properties and interpreted regarding the specific models.

The same derivation process (translation, reduction or extension) holds for the properties (P') expressed with respect to specific models; either they are the translation of the global properties (P) of the reference model, or they are tightly related to the specific models, or they are extended form of the reference model properties. Note that in the two last cases, the properties are more constraining than the initial ones.

Consequently, according to the relation between a reference model (M) and the specific ones (M'), one can interpret the performed analysis. This is summarised by laws of the following form:

Let $\mathcal{R}_{m\,d}$ stands for the:

- either $\mathcal{R}_{m\,t}(M, M')$:*translation/adaptation* of the reference model,

Translation	transfer of unproved properties from M' to M transfer of proved properties from M' to M
Reduction	transfer of unproved properties from M' to M
Extension	transfer of proved properties from M' to M

Table 1. Analysis interpretation laws

- or $\mathcal{R}_{m\,r}(M, M')$: *Reduction* of the reference model,
- or $\mathcal{R}_{m\,e}(M, M')$: *Extension* of the reference model.

Let $\mathcal{R}_{p\,d}$ be the :

- *Explicitation* (translation, reduction, extension) of the relationship between P and P':

$$\frac{M \vDash P \quad \mathcal{R}_{m\,d}(M, M') \quad M' \vDash P' \quad \mathcal{R}_{p\,d}(P, P')}{M \ \vDash_{(orNot)} \ P'}$$

In the case of reduction, the properties which are not proved on the reduced model are also not proved for the reference model. In the case of extension, the properties proved on the extended models are also proved for the reference model; the impact of the laws on the analysis of properties is summarised in the table Tab. 1

Properties of Heterogeneous Specification The heterogeneous aspect in a system may appears through its properties. Some properties may be well expressed and analysed with a particular method rather than another one. Therefore, increasing a reference model with specific properties expressed in another formalism is a typical case of derivation that leads to multifacet analysis since the added properties will require appropriate analysis tools.

Our approach of multifacet analysis is suited for covering the specification of the various facets of a system with dedicated languages and their analysis with appropriate tools. It is also suited for combining reasoning tools in the same development or analysis project. More specifically, multifacet analysis favours the combined use of theorem-proving and model-checking tools to analyse system properties.

In the following we consider an illustrative case where both reasoning techniques (theorem-proving and model-checking) are useful. We will build from an abstract model, a specific one which is the extension of the former by properties which are written in another language.

4. MANET Specification and Analysis

The study of MANET (Mobile Ad-hoc Network)[12] is an active and challenging field as this type of network is more and more growing and supporting small and medium size applications such as mobile services sharing, wireless peer-to-peer systems, etc.

We build a reference model with Event B abstract system. We derive a specific model from the latter by extending it with temporal (LTL) properties. The overall system is analysed with two different tools B4free[3] and ProB[4][15].

We chose the field of MANET for this study because it is a challenging field in the frontier of computer networks and software engineering. Especially, communication protocols, which are specific software systems, should be correct to ensure the (quality of) services deployed on networks.

From the software system point of view, the nature and properties of MANET (dynamicity, mobility, correctness, etc) need a multifacet specification and the combined use of various reasoning tools. Indeed handling dynamic behaviour of processes and their architecture is not well treated with standard approaches such as Finite State Machine ones; but event-based approaches provide solutions. Moreover, the properties related to MANET may be expressed with temporal logic based formalisms. Accordingly we combine in our study an event-based approach and a model-checking approach which permits the expression of temporal properties.

We present in the following the main features of MANET, the tools used to handle our study and the performed property analysis.

4.1. Overview of Mobile Ad-hoc Network

A mobile ad-hoc network (MANET) [12] is a network formed by wireless mobile nodes (called ad-hoc nodes) which are the users equipments or devices. A MANET has no dedicated network infrastructure, but each node serves as a part of the network and acts as a *router* to forward messages or packets since there is no router dedicated to that task.

A mobile ad-hoc network is formed only when a group of users put together their resources to enable and perform communications; hence a mobile ad-hoc network is dynamically created and may also desappears quickly.

In a MANET, the nodes communicate either by exchanging directly or via intermediate nodes. Technically they use ISM band[5] and more generally Wireless Lan technologies. Each node is equipped with one or more radio interfaces with specific transmission features. The *transmission range* of a node is the transmission area accessible from this node. All the nodes in this range are accessible directly (one hop);

3 B4free is one of the tool dedicated to Event B: `www.B4free.fr`

4 ProB `www.stups.uni-duesseldorf.de/ProB/`, is a public domain model-checker for B.

5 they are radio system frequency initially dedicated to industrial, scientific and medical usage.

they are called the neighbours. To address a known node which is not in its transmission range, the sender node sends its packet to one of the neighbour nodes which is closer to the destination node (according to the transmission ranges). Each node may communicate directly or indirectly using relay nodes (multi-hop), with other nodes that are outside the sender range. A message or packet sent to a node reaches it unless the net is partitionned.

Dynamic Aspect. One of the main features of a MANET is its dynamic aspect: the structure or topology of the network is frequently changing. A node may join or leave the net at any time, changing the net topology. The structure or topology of the net is then highly dynamic.

Mobility Aspect. The ad-hoc nodes may move at any time and very frequently due to their mobile nature; consequently this impacts not only on the net topology but also on its quality; there may be route changes, information loss, partitions of the network into different networks, etc. As far as routing is concerned, in classical infrastructure-based network, there are one or several nodes called routers that are in charge of routing packets between nodes. For this purpose the routers and the nodes are equipped with a routing table where there is the information about how to join a given destination node or a network identified with an Internet Address (IP Address).

In the scope of MANET, efficient routing protocols development is a challenging concern.

Concerning the time, it is assumed to be discrete and divided into frames. A node has a set of neighbour nodes during a frame. During a frame a node may be idle, it also may send messages, receive messages, forward the received messages.

Before sending a message to a destination, a source node sn which does not have the destination node address, sends a route request to get this destination address. The request travels through the net possibly with multi-hop and reaches the destination which sends back it address. When the address is received by sn the latter can send its message to the right destination address.

4.2. The Used Tools: Event B and ProB

In this study we chose Event B as a practical framework for the specification of the MANETs. Prior to the formal specification itself we provide an overview of the Event B approach.

4.2.1. Overview of Event B Within the Event B framework, asynchronous systems may be developed and structured using *abstract systems* [2, 4]. *Abstract systems* are the basic structures of the so-called *event-driven* B, and they replace the *abstract machines* which are the basic structures of the earlier *operation-driven* approach of the B method[1]. An *abstract system* [2, 4] describes a mathematical model of a system behaviour[6]. It is made mainly of a state description (constants, properties, variables and invariant) and several *event* descriptions. Abstract systems are comparable to Action Systems [8]; they describe a nondeterministic evolution of a system through guarded actions. Dynamic constraints can be expressed within abstract systems to specify various liveness properties [4, 11]. The state of an abstract system is described by variables and constants linked by an invariant. Variables and constants represent the data space of the system being formalised. Abstract systems may be refined like abstract machines [11, 3].

Data of an Abstract System. At a higher level an abstract system models and contains the data of an entire model, be it distributed or not. Abstract systems have been used to formalise the behaviour of various (including distributed) systems [2, 10, 11, 3]. Considering a global vision, the data that are formalised within the abstract system may correspond to all the elements of the distributed system.

Events of an Abstract System. Within B, an event is considered as the observation of a system transition. Events are spontaneous and show the way a system evolves. An event e is modelled as a *guarded substitution*: $e \mathrel{\widehat{=}} eG \Longrightarrow eB$ where eG is the event *guard* and eB the event *body* or *action*.
An event may occur or may be observed only when its guard holds. The action of an event describes with generalised substitutions how the system state evolves when this event occurs. Several events may have their guards held simultaneously; in this case, only one of them occurs. The system makes internally a nondeterministic choice. If no guard is true the abstract system is blocking (deadlock).

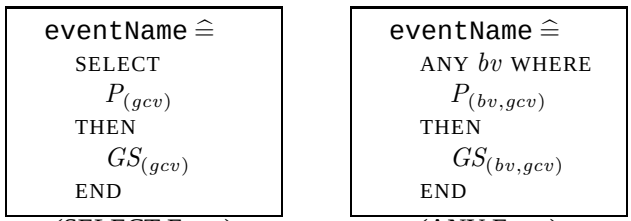

Figure 2. General forms of events

An event has one of the general forms (Fig. 2) where gcv denotes the global constants and variables of the abstract system containing the event; bv denotes the bound variables (variables bound to ANY). $P_{(bv,gcv)}$ denotes a predicate P expressed with the variables bv and gcv; in the same

6 A system behaviour is the set of its possible transitions from state to state beginning from an initial state

way $GS_{(bv,gcv)}$ is a generalised substitution S which models the event action using the variables bv and gcv. The SELECT form is a particular case of the ANY form. The guard of an event with the SELECT form is $P_{(gcv)}$. The guard of an event with the ANY form is $\exists(bv).P_{(bv,gcv)}$.

Semantics and Consistency. The semantics of an abstract system relies on its invariant and is guaranteed by proof obligations(POs). The *consistency* of the model is established by such proof obligations: *i) the initialisation U should establish the invariant I*: $[U]I$;
ii) each event of the given abstract system should preserve the invariant of the model. The proof obligation of an event with the ANY form (Fig. 2) is:

$$I_{(gcv)} \wedge P_{(bv,gcv)} \wedge \mathsf{term}(GS_{(bv,gcv)}) \Rightarrow [GS_{(bv,gcv)}]I_{(gcv)}$$

where $I_{(gcv)}$ stands for the invariant of the abstract system; the predicate $\mathsf{term}(GS_{(bv,gcv)})$ expresses that the event should terminate. The deadlock-freeness should be established for an abstract system: the disjunction of the event guards should be true. The event-based semantics of an abstract system A is the event traces of A ($traces(A)$); the set of finite event sequences generated by the evolution of A. The B method is supported by the industrial theorem provers Atelier-B [13] and B-Toolkit [7] and also by B4free.

4.2.2. Overview of ProB The ProB tool [15, 16] is an animator and a model checker for B specifications. It provides functionalities to display graphical view of automata. It supports automated consistency checking of B specifications (an abstract machine or a refinement with its state space, its initialisation and its operations). The consistency checking is performed on all the reachable states of the machine. The ProB also provides a constraint-based checking; with this approach ProB does not explore the state space from the initialisation, it checks whether applying one of the operation can result in an invariant violation independently from the initialisation.

The ProB offers many functionalities. The main ones are organised within three categories: *Animation*, *Verification* and *Analysis*. Several functionalities are provided for each category but, we just list a few of them which are used in this article.

The *Animation* category gathers the following functionalities:
Random Animation: it starts from an initial state of the abstract machine and then, it selects in a random fashion one of the enabled operations, it computes the next state accordingly and proceeds the animation from this state with one of the enabled operations;
View/Reduced Visited States: it displays a minimised graph of the visited states after an animation;
View/Current State: it displays the current state which is obtained after the animation.

In the *Verification* category, the following functionalities are available:
Temporal Model Checking: starting from a set of initialisation states (initial nodes), it systematically explores the state space of the current B specification. From a given state (a node), a transition is built for each enabled operation and it ends at a computed state which is a new node or an already existing one. Each state is treated in the same way.
LTL Model Checking: this functionality enables one to check the specification against a given LTL property.
Constraint Based Checking: it checks for invariant violation when applying operation independently from initialisation states.

In the *Analysis* category we have the following functionalities:
Compute Coverage: the state space (the nodes) and the transitions of the current specification are checked, some statistics are given on deadlocked states, live states[7], covered and uncovered operations.
Analyse Invariant: it checks if some parts of the current invariant are true or false;

The ProB tool is used in our study to help in proving consistency (invariant violation) and to check liveness properties. We express the desired liveness properties with the ProB dedicated LTL formalism. In the case where the consistency is not completely achieved the ProB tool helps to discover the faults by providing an explicite state that violates the invariant. Then the proof process is fast.

4.3. Formal Specification of MANET

In our study, a MANET is viewed as an evolving global system. Formally, it is a set of nodes with a connection relationship: a configuration. The evolution of the MANET is viewed as a sequence of configurations; going from a configuration to another is observed as an event and it depends on the actions performed by the net nodes.

Therefore a formal specification of a MANET is a set of possible sequence of configuration of the considered nodes. We describe the configuration by *state variables* (hence a state space); the sequence of configurations is modelled through the enabling of *events* which possibly modify the state space.

4.3.1. Event B Specification of MANETs The MANET is formed from nodes; each node has some features: an identifier, a location, an IP address, a connection relation that indicates its neighbours, etc.

We have considered two aspects in the Event B specification of MANET: the structuring of the networks (the configuration related to the net topology) and the routing in the

7 the already computed states.

Event	Description
newRange	A new network range appears
joinRange	A node joins a range
leaveRange	A node leaves a range
newNode	A new node appears
newMsg	A node initiate a message

Table 2. Network structuring events

networks. Concerning the structuring we deal with the creation of a network by nodes which have a given range, other nodes may join or leave this range. Therefore we link the range of a node with a given abstract network. This network is dynamic, the nodes leave and join it at any time, new ranges appear, others disappear, etc.

As far as routing is concerned we consider one of the widely studied routing protocol of MANET: *Ad-hoc On demand Distance Vector* (AODV) [12].

Therefore a part of our B specification is related to the structuring and another one deals with the routing.

Specifying the MANET Structure The structuring of a MANET is achieved using a set of state variables and an invariant that describes the nodes and their current configurations ($nodes \subseteq NODE, ranges \subseteq RANGE, rangNodes \in ranges \leftrightarrow nodes, \cdots$) ; the evolution depends on a set of events that we have identified: first the observation of a net creation: *newRange*; an existing net may disappear if there is no more connected nodes: *rmvRange*. The other events considered for the network structuring are summarised in the table Tab. 2;

Accordingly, a node is modelled as a process. A set of events defines the process behaviour which leads the evolution of the system. A node process may initiate a message for a given destination, send a message, receive a message, forward a message, leave a net (a transmission range). This behaviour is observed only when a net exists; that means the net structuring events are related to those needed for the routing. The combination of the two categories of events forms an abstract MANET specification which is a reference model for the specification. It describes a system composed of node processes and abstract MANET networks.

Specifying the AODV routing protocol Within the Adhoc On demand Distant Vector (AODV) protocol, each node acts as a router, contributes to construct routes and forward messages to other nodes. There are two phases of the protocol: route discovery and route maintenance. Route discovery is achieved by exchanging Route Request (RREQ) and Route Response (RREP) messages. The algorithm of the nodes is as follows: when a node desires to set up a route to a destination node, it broadcasts a RREQ message to its neighbours (the node in its range). The RREQ/RREP mes-

Event	Description
sndRREQ	Route Request sending
fwdRREQ	Route Request forwarding
rcvRREQ	Route Request receiving
sndRREP	Route Response sending
fwdRREP	Route Response forwarding
rcvRREP	Route Response receiving

Table 3. Routing events

sages have the following main parameters: the source node Id, the destination node Id, the number of hop.

When a node nd receives a RREQ message, *i)* either nd is itself a destination and nd responds with a RREP or it is an active route to the searched destination node then nd responds with a route information using the RREP message; *ii)* otherwise nd broadcasts the RREQ further with the hop count of RREQ increased by 1. When a node nd receives a duplicate RREQ, it drops the message. The routing of message is symmetric when a node receives a RREP message.

Our Event B specification comprises the events related to the routing protocol described above. These events are listed in the table Tab. 3.

The B specification of a MANET, the reference model, is then an abstract system equipped with these events (see Fig. 3).

We give in the following the specification of the `sndRREQ` event to illustrate the principle. Here, any node (sn) may send a message (msg) that it has already prepared ($msg \in reqMsg[\{sn\}]$) to another node in its range ($otherNodesInRange$). Message exchanges are modelled using abstract channels ($inRepMsg$,$repMsg$) which are sets.

```
sndRREQ ≙  /* route request from sn to dn */
ANY sn, msg  WHERE
   sn ∈ nodes /* source */
   ∧ msg ∈ MSG ∧ msg ∈ messages
   ∧ msg ∈ reqMsg[{sn}] /* a msg initiated by nd */
THEN
   LET otherNodesInRange
   BE otherNodesInRange = {ndi | ndi ∈ nodes
   ∧ ndi ≠ sn ∧ rangNodes⁻¹(sn) = rangNodes⁻¹(ndi)}
   IN   inReqMsg :=
          inReqMsg ∪ (otherNodesInRange ∗ {msg})
      ||  reqMsg := reqMsg − {(sn ↦ msg)}
   END
END
```

4.3.2. Consistency and Refinement of System The first abstract system is proved consistent (see Sect.4.2.1) using the B4free tool. Then it is refined; more details are added to the state space and the event specifications; for instance we consider the management of the IP addresses for the nodes and exchanged messages. Unlike in the abstract sys-

SYSTEM $MANET$
SETS NODE, RANGE, MSG /* abstract sets */
VARIABLES
$nodes, ranges, messages, \cdots$ /* state variables*/
INVARIANT
$nodes \subseteq NODE$
$\wedge$ $ranges \subseteq RANGE$
$\wedge$ $messages \subseteq MSG$
$\wedge$ $rangNodes \in ranges \leftrightarrow nodes$
$\wedge$ $\cdots$
INITIALISATION
$nodes, ranges, messages, rangNodes := \emptyset, \emptyset, \emptyset, \emptyset$
$\|$ $\cdots$
EVENTS
newNODE $\widehat{=}$ $\cdots$
; newRANGE $\widehat{=}$ $\cdots$
; joinRange $\widehat{=}$ $\cdots$
; leaveRange $\widehat{=}$ $\cdots$
; newMsg $\widehat{=}$ $\cdots$
; sndRREQ $\widehat{=}$ $\cdots$
; rcvRREQ $\widehat{=}$ $\cdots$
; fwdRREQ $\widehat{=}$ $\cdots$
; newRespMsg $\widehat{=}$ $\cdots$
; sndRREP $\widehat{=}$ $\cdots$
; rcvRREP $\widehat{=}$ $\cdots$
END

Figure 3. Structure of the Abstract System

tem where a packet destination is nondeterministically selected, in the refinement the nodes and the messages have IP addresses, therefore, the receiver node is checked against the destination IP address. The resulting refined system is proved correct with respect to consistency using the Event B tool. However to accomplish the proofs, we combine the use of B4free and ProB; both tools use the B abstract system as a common input formalism; then a multifacet approach based on this common input and specific tools helps to prove the specification. That is, when a proof obligation is not discharged by B4free, we model-check the specification and discover possible errors by displaying and analysing the error state. Accordingly the feedback is propagated on the common input and we iterate. The multifacet analysis approach also helps here to precise the correct ordering of the events: the simulation functionalities and the listing of uncovered operations help to correct the B abstract system. This aspect is very important because, an abstract system proved correct, may have an incomplete or wrong behaviour; for example if we have an event which is never enabled. Using the multifacet approach, helps us to get a complete analysis.

The following table shows a ProB experiment result; one deadlock is detected after the exploration of 31257 nodes and 1168 transitions ; all operations (the B events) are covered, with the occurrences indicated in the table.

NODES		
invariant_violated	:	0
deadlocked	:	1
live	:	2521
explored_transitions	:	1168
open	:	28735
total	:	31257
TOTAL_OPERATIONS		
44110		
COVERED_OPERATIONS		
initialise_machine	:	1
newRANGE	:	225
rcvRREP	:	14
sndRREP	:	29
newRespMsg	:	300
sndRREQ	:	1829
rcvRREQ	:	1697
newNODE	:	10487
joinRange	:	7411
leaveRange	:	9721
newMsg	:	11042
fwdRREQ	:	1354
UNCOVERED_OPERATIONS		

The state corresponding to the deadlock is carefully analysed. We discover that it corresponds to a situation where there are nodes with some packets to be transmitted but no node in the current net range. This corresponds to a real-life situation which is due to the dynamic aspect of the MANET and the mobility of nodes. To confirm our hypothesis a feedback is then propagated on the Event B specification. The model is corrected by strengthening the guard of message initiation by the hypothesis of non-emptiness of the net range. Thus the analysis of the model runs without errors[8]. In the real-life situation, this corresponds to the fact that after a while the net may be reconstituted with other nodes.

4.4. Analysis of MANET properties

Many properties of the MANET routing protocol are well-expressed using LTL formula which is not supported by the Event B and the B4free tool. We express these liveness properties with the LTL formalism of the ProB tool. Then we extend the Event B abstract system with these LTL properties, the result is a specific model used to perform the analysis of the liveness properties viewed as a facet.

Liveness Properties Analysis A part of the properties that have been expressed and checked with ProB are listed here. The syntax is the standard one of LTL except $e(eventName)$ which expresses the enabling of an event. The properties (P') help to establish the correcteness of our extended formal model (already equipped with the invariant properties (P).

8 the experiment result tables, not displayed here, show 0 deadlocked states for hundreds of explored states and transitions.

$\mathbf{P}_1$. After a route request, a response message is initiated: $G(e(sndRREQ) \Rightarrow F(e(newRespMsg)))$ false

$\mathbf{P}_2$. A response message initiated by a node is finally sent: $G(e(newRespMsg) \Rightarrow F(e(sndRREP)))$ true

$\mathbf{P}_3$. A route request is always followed by a response: $G(e(sndRREQ) \Rightarrow F(e(sndRREP)))$ false

$\mathbf{P}_4$. A route request may be followed by a response: $e(sndRREQ) \Rightarrow F(e(sndRREP))$ true

$\mathbf{P}_5$. A route request may be finally received: $F(e(sndRREQ) \Rightarrow X(e(rcvRREQ)))$ true

Consequently, the extended model (M') is checked with respect to P and P'. In the current case where the input specifications are the same, we come to the conclusion that $M' \vDash P'$ by applying the interpretation laws (see Sect. 3.3). Formally our model of the MANET extended with the stated properties, is correct with respect to these properties.

5. Conclusion

We have presented an approach to master the specification and the analysis of heterogeneous system. The approach is based on the one hand on the combined use of a reference abstract model and specific models which are derived from the former in order to preserve the semantics; on the other hand the approach is based on the combined use of tools appropriate to various facets of the system being studied. For scalability, a reference model expressed with a (typed) FOL and a state/event-based dynamic aspect seems appropriate. We illustrate the work with an experiment on the specification and the verification of Mobile Ad-hoc Network (MANET) system which involves various facets such as network structuring, routing and correctness. In this case a B model is built and checked as a reference model using the B4free tool; it is then extended to get a specific model which is further analysed with ProB. A part of this work is related to works on the combination and the translation of specifications [14, 5]. However, in our knowledge the multifacet approach to deal with heterogeneous specification and verification is new. There are may works on the challenges posed by MANET, they mainly address the security and performance issues [12]. Further works are planed to mechanically assist, even partially, the derivation of the specific models in order to make it easy the update of the reference model and the other specific ones, according to the feedback from the analysis of the models.

References

[1] J-R. Abrial. *The B Book*. Cambridge University Press, 1996.

[2] J-R. Abrial. Extending B without Changing it (for developping distributed systems). *Proc. of the 1st Conf. on the B method, H. Habrias (editor), France*, pages 169–190, 1996.

[3] J-R. Abrial, D. Cansell, and D. Mery. Formal Derivation of Spanning Trees Algorithms. In D. Bert et al., editor, *ZB'2003 – Formal Specification and Development in Z and B*, volume 2651 of *LNCS*, pages 457–476. Springer-Verlag, 2003.

[4] J-R. Abrial and L. Mussat. Introducing Dynamic Constraints in B. In *Proc. of the 2nd Conference on the B method, D. Bert (editor)*, volume 1393 of *Lecture Notes in Computer Science*, pages 83–128. Springer-Verlag, 1998.

[5] Y. Aït Ameur, R. Delmas, and V. Wiels. A Framework for Heterogeneous Formal Modeling and Compositional Verification of Avionics Systems. In *MEMOCODE*, pages 223–232. IEEE, 2004.

[6] C. Attiogbé. Practical Combination of Theorem Proving and Model Checking for the Multi-facet Analysis. In *Proc. of SOFSEM'05*, pages 1–10, 2005.

[7] B-Core. *B-Toolkit, www.b-core.com*. UK, consulted in 2007.

[8] R.J. Back and R. Kurki-Suonio. Decentralisation of Process Nets with Centralised Control. In *Proc. of the 2nd ACM SIGACT-SIGOPS Symp. on Principles of Distributed Computing*, pages 131–142, 1983.

[9] R. Boulton, A. Gorgon, M.J.C. Gordon, J. Hebert, and J. van Tassel. Experience with Embedding Hardware Description Language in HOL. In *Proc. of the International Conference on Theorem Provers in Circuit Design: Theory, Practice and Experience*, pages 129–156, North-Holland, 1992. IFIP TC10/WG 10.2.

[10] M. Butler and M. Walden. Distributed System Development in B. *Proc. of the 1st Conference on the B method, H. Habrias (editor), France*, pages 155–168, 1996.

[11] D. Cansell, G. Gopalakrishnan, M. Jones, and D. Mery. Incremental Proof of the Producer/Consumer Property for the PCI Protocol. In D. Bert, J. P. Bowen, M. C. Henson, and K. Robinson, editors, *ZB'2002 – Formal Specification and Development in Z and B*, volume 2272 of *LNCS*, pages 22–41. Springer-Verlag, 2002.

[12] I. Chlamtac, M. Conti, and J. Liu. Mobile Ad hoc Networking: Imperatives and Challenges. *Ad Hoc Networks*, 1(1):13–64, July 2003.

[13] ClearSy. *Atelier B V3.6, www.clearsy.com*. Steria, Aix-en-Provence, France, consulted in 2007.

[14] S. Katz and O. Grumberg. A Framework for Translating Models and Specifications. In *Proceedings of Integrated Formal Methods (IFM'2002)*, volume 2335 of *LNCS*, pages 145–164, UK, May 2002. Springer-Verlag.

[15] M. Leuschel and M. Butler. ProB: A Model Checker for B. In Keijiro A., Stefania G., and Dino M., editors, *FME 2003: Formal Methods Europe*, LNCS 2805, pages 855–874. Springer-Verlag, 2003.

[16] M. Leuschel and E. Turner. Visualizing Larger State Spaces in ProB. In *Proc. of ZB'05*, volume 3455 of *LNCS*, pages 6–23. Springer-Verlag, April 2005.

[17] R. Milner. *A Calculus of Communicating Systems*. Springer-Verlag, Berlin, 1980.

[18] T. Murata. Petri-Nets: Properties, Analysis and Applications. In *Proc. IEEE*, volume 77, pages 541–580. IEEE, 1989.

[19] D.M.R. Park. Concurrency and Automata on Infinite Sequences. In P. Deussen, editor, *5th GI Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, 1981.